REMARKS

This Application has been carefully reviewed in light of the Office Action dated December 31, 2007 (the "*Office Action*"). At the time of the *Office Action*, Claims 40-73 were pending in the Application with Claims 1-39 being withdrawn. The Examiner rejects Claims 40-51, 60-63, 65-68, and 70-73, and allows Claims 52-59, 64, and 69. Applicants amend Claims 40, 48, 49, 60-65, and 72; and add new Claim 74. Applicants submit that no new matter is added by these amendments or added claims. Applicants respectfully request reconsideration and favorable action in this case.

Allowable Subject Matter

Applicants note with appreciation the Examiner's indication that Claims 52-59, 64, and 69 are allowed. Claims 52-59, 64, and 69 have not been amended and, therefore, remain in condition for allowance.

Pursuant to 37 C.F.R. § 1.104, Applicants respectfully issue a statement commenting on the Examiner's reasons for allowance. Applicants respectfully disagree with the Examiner's reasons fro allowance to the extent that they are inconsistent with applicable case law, statutes, and regulations. Furthermore, Applicants do not admit to any characterization or limitation of the claims, particularly any that are inconsistent with the language of the claims considered in their entirety and including all of their constituent limitations or to any characterization of a reference by the Examiner.

Section 101 Rejections

The Examiner rejects Claims 60-64 under 35 U.S.C. § 101 on the basis that the claims recite software *per se*.

As an initial matter, Applicants do not necessarily agree that the systems recited in Claims 60-64 are software per se. The authority cited by the Examiner merely states that ""[s]oftware per se" is non-statutory under 35 USC 101 because it is merely a set of instructions without any defined tangible output or tangible result being produced." (*Office Action*, page 3, citing *State Street Bank Trust Co. v. Signature Financial Group Inc.*, 149 Fed 1368, 47 USPQ2d 1596 (Fed. Cir. 1998)). Applicants respectfully submit that each of Claims 60-64 produce a useful, concrete, and tangible result and, thus, are directed to statutory subject matter. However, Applicants have made clarifying amendments to Claims 60-64 in

order to address the Examiner's concerns. For example, Applicants have amended independent Claims 60 and 64 to recite "a processing system" rather than "a software module." Applicants have amended dependent Claims 61-63 to recite "the processing system" rather than "the software module." For at least these reasons, Applicants respectfully submit that Claims 60-64 do not recite software *per se* and are indeed directed to statutory subject matter under 35 U.S.C. § 101.

Accordingly, Applicants respectfully request that the 35 U.S.C. § 101 rejections of Claims 60-64 be withdrawn.

Section 103 Rejections

The Examiner rejects Claims 40-42, 46-51, 60-62, and 65-67 under 35 U.S.C. § 103(a) as being unpatentable over U.S. Patent No. 5,586,328 issued to Caron et al. ("*Caron*") in view of U.S. Patent No. 6,199,063 issued to Colby et al. ("*Colby*"), and further in view of WO 92/15066 issued to Ming-Chien Shan dated 2/26/91 ("*Shan*"). The Examiner rejects Claims 43-45, 51, 63, and 68 under 35 U.S.C. § 103(a) as being unpatentable over *Caron* in view of *Colby* and further in view of *Shan* and in view of U.S. Patent No. 5,926,819 issued to Doo et al. ("*Doo*"). The Examiner rejects Claims 70-73 under 35 U.S.C. § 103(a) as being unpatentable over *Caron* in view of *Colby* and further in view of *Shan* and in view of U.S. Patent No. 5,805,804 issued to Laursen et al. ("*Laursen*"). Applicants request reconsideration and allowance of Claims 40-51, 60-63, and 65-68, and 70-73 for the reasons discussed below.

A. Claims 40, 43-47, 51, 60, 63, 65, and 68

Independent Claim 40 of the present Application, as amended, recites:

> A method of generating dependency information for code objects stored in a database, comprising:
>
> recursively querying a database for one or more dependencies of procedural code objects stored in the database;
>
> identifying one or more dependencies of procedural code objects stored in the database;
>
> generating a dependency information tracking array based on the identification of one or more dependencies of procedural code objects; and

> stopping the recursive query of the database upon identifying a dependency that is already included in the dependency information tracking array.

Applicants respectfully submit that independent Claim 40 is allowable over the proposed *Caron-Colby-Shan* combination since the references, whether considered alone or in combination, do not disclose, teach, or suggest the particular combination of elements recited in Applicants' claim.

For example, the proposed *Caron-Colby-Shan* combination fails to disclose, teach, or suggest "stopping the recursive query of the database upon identifying a dependency that is already included in the dependency information tracking array," as recited in Claim 40. In the *Office Action*, the Examiner acknowledges that neither *Caron* nor *Colby* disclose "recursively querying a database." (*Office Action*, page 5). Since *Caron* and *Colby* do not disclose the recursive querying of a database, it follows that neither *Caron* nor *Colby* can be said to disclose, teach, or suggest "stopping the recursive query of the database upon identifying a dependency that is already included in the dependency information tracking array," as recited in Claim 40.

Applicants further submit that the disclosure of *Shan*, which is relied upon by the Examiner for disclosure of the recursive querying of a database, does not cure the deficiencies of *Caron* and *Colby*. Rather, *Shan* merely discloses that a recursive query "can be described as a query which queries itself." (*Shan*, page 9). For example, *Shan* discloses that "a query of the form "FIND GRANDPARENTS OF [X] . . . can be recursively derived from the information in the database, for example, by a query of the form "FIND PARENTS OF [FIND PARENTS OF [X]]"." (*Shan*, page 9). According to *Shan*, "[r]elational algebra provides a powerful technique for optimizing the evaluation of database queries, but recursive queries have not been amenable to such optimization techniques." (*Shan*, page 12). Accordingly, *Shan* discloses "a novel fixpoint operator and new transformation procedures . . . for translating a recursive query into a relational algebra expression and then simplifying the expression." (*Shan*, page 12). Specifically, *Shan* discloses that "a method of evaluating a recursive query of a database 11 comprises translating a recursive query into an expression that includes a fixpoint operator, as indicated by a "translate using []" process circle 13; optimizing the expression according to a set of transformation procedures as indicated by an "optimize" process circle 15; and evaluating the optimized expression, as indicated by an

"evaluate" process circle 17, by reference to data in the database 11." Thus, *Shan* merely discloses a method for translating a recursive query into a relational algebra expression. *Shan* does not disclose, teach, or suggest "stopping the recursive query of the database upon identifying a dependency that is already included in the dependency information tracking array," as recited in Claim 40.

As another example, the proposed *Caron-Colby-Shan* combination fails to disclose, teach, or suggest "recursively querying a database for one or more dependencies of procedural code objects stored in the database," as recited in Applicants' independent Claim 40. In the *Office Action*, the Examiner relies upon *Caron* for disclosure of "dependencies of procedural code objects stored in a database" and on *Colby* for disclosure of querying a database. The Examiner acknowledges, however, that neither *Caron* nor *Colby* disclose "recursively querying a database" and instead relies upon *Shan* for disclosure of the recited operation. (*Office Action*, page 5). Thus, to reject Applicants' step of "recursively querying a database for one or more dependencies of procedural objects stored in the database," the Examiner must piece together the disclosures of three separate references. Even if the cited references disclose the elements alleged by the Examiner (which Applicants do not admit), such a piecemeal rejection of Applicants' claim language fails to give credence to the particular combination of features recited in Applicants' claim. Applicants respectfully submit that a rejection of Claim 40 under the *Caron-Colby-Shan* combination, in the manner provided by the Examiner, can only result from piecing together disjointed portions of unrelated references to reconstruct Applicants' claims with the benefit of hindsight.

The Federal Circuit has made clear that is improper for an Examiner to use hindsight having read the Applicant's disclosure to arrive at an obviousness rejection. *In re* Fine, 837 F.2d 1071, 1075, 5 U.S.P.Q.2d 1596, 1600 (Fed. Cir. 1988). It is improper to use the claimed invention as an instruction manual or template to piece together the teachings of the prior art so that the claimed invention is rendered obvious. *In re* Fritch, 972 F.2d 1260, 23 U.S.P.Q.2d 1780 (Fed. Cir. 1992). In this case, Applicants respectfully submit that the Examiner has used Applicants' claimed invention as an instruction manual to combining the teachings of *Caron*, *Colby*, and *Shan* to conclude that the references disclose "recursively querying a database for one or more dependencies of procedural code objects stored in the database." Because the hindsight reconstruction of Applicants' claim is improper, Applicants submit that Claim 40 is allowable over the proposed *Caron-Colby-Shan* combination.

For at least these reasons, Applicants respectfully request consideration and allowance of Claim 40, together with Claims 43-47 and 51 that depend on Claim 40.

The Examiner also relies on the proposed *Caron-Colby* combination to reject independent Claims 60 and 65. Applicants respectfully submit, however, that the proposed references do not disclose, teach, or suggest the elements recited Applicants' independent Claims 60 and 65. For example, Claim 60 recites "a software module operable to . . . recursively query the database for one or more dependencies of procedural code objects stored in the database . . . and stop the recursive query of the database upon identifying a dependency that is already included in the dependency information tracking array." As another example, Claim 65 recites "[a] computer-readable medium encoded with logic operable, when executed on a computer processor, to . . . recursively query the database for one or more dependencies of procedural code objects stored in the database . . . and stop the recursive query of the database upon identifying a dependency that is already included in the dependency information tracking array." Thus, for reasons similar to those discussed above with regard to Claim 40, Applicants respectfully submit that Claims 60 and 65 are allowable over the proposed *Caron-Colby* combination. Applicants respectfully request reconsideration and allowance of Claims 60 and 65, together with Claims 63 and 68 that depend on Claims 60 and 65, respectively.

B. Claim 72

Claim 72 has been rewritten in independent form to include the limitations recited in Claim 1 prior to any amendment in this Response to Office Action. Accordingly, independent Claim 72 recites:

> A method of generating dependency information for code objects stored in a database, comprising:
>
> recursively querying a database for one or more dependencies of procedural code objects stored in the database;
>
> identifying one or more dependencies of procedural code objects stored in the database;
>
> generating a dependency information tracking array based on the identification of one or more dependencies of procedural code objects; and
>
> for each of the one or more dependencies identified, determining whether the dependency already occurs in the graph; and

terminating the recursive query of the database upon determining that one of the one or more dependencies already occurs in the graph.

Applicants respectfully submit that independent Claim 72 is allowable over the proposed *Caron-Colby-Shan-Laursen* combination since the references, whether considered alone or in combination, do not disclose, teach, or suggest the particular combination of elements recited in Applicants' claim.

For example, the proposed *Caron-Colby-Shan* combination fails to disclose, teach, or suggest "for each of the one or more dependencies identified, determining whether the dependency already occurs in the graph" and "terminating the recursive query of the database upon determining that one of the one or more dependencies already occurs in the graph," as recited in Claim 72. In the *Office Action*, the Examiner acknowledges that *Caron*, *Colby*, and *Shan* do not disclose the recited claim elements. (*Office Action*, page 14). Instead, the Examiner relies on *Laursen* for disclosure of the recited claim elements. Specifically, the Examiner states that "*Laursen* teaches directed graph and tracking arrays (dext 99, 101, 121, and 125)." (*Office Action*, page 14). Thus, the Examiner identifies only that *Laursen* relates to directed graph and tracking arrays but does not identify that *Laursen* actually teaches "for each of the one or more dependencies identified, determining whether the dependency already occurs in the graph" and "terminating the recursive query of the database upon determining that one of the one or more dependencies already occurs in the graph," as recited in Claim 72. Additionally, Applicants are not sure what "dext" refers to and have determined that there is no reference in *Laursen* to the reference numerals 99, 101, 121, and 125. It does not appear that the reference numerals refer to page numbers, line numbers, or paragraph numbers. In short, the Examiner's rejection of the claim language leaves Applicants' guessing as to the basis fro the Examiner's rejection and provides the Applicants with little guidance in how to respond.

Furthermore, Applicants have performed a review of *Laursen* and have concluded that *Laursen* does not at all relate to recursive queries of a database. There is no discussion in *Laursen* of querying a database, no mention of the word "recursive" as used in the context of a query, and certainly no discussion of terminating or stopping a recursive query. Rather, *Laursen* merely discloses a system that "allows applications to be split such that client devices (set-top boxes, personal digital assistants, etc.) can focus on presentation, while backend services running in a distributed server complex, provide access to data via

messaging across an abstracted interface." (*Laursen*, Column 2, lines 19-24). Specifically, *Laursen* discloses:

> Each service provides access to a particular type of data or resources. A service exports one or more functions, which perform specific actions related to the data or resource. A client program invokes a function by communicating with the service that exports that function.

(*Laursen*, Column 2, lines 24-29). Thus, while *Laursen* relates generally to a method and system for retrieving data, *Laursen* merely discloses splitting the operations of retrieving data between multiple services based on the types of data. *Laursen* does not disclose, teach, or suggest "for each of the one or more dependencies identified, determining whether the dependency already occurs in the graph" and "terminating the recursive query of the database upon determining that one of the one or more dependencies already occurs in the graph," as recited in Claim 72. Accordingly, Claim 72 is allowable over the proposed *Caron-Colby-Shan-Laursen* combination.

As another example, Applicants note that Claim 72 also recites "recursively querying a database for one or more dependencies of procedural code objects stored in the database." With regard to Claim 40, from which Claim 72 previously depended, the Examiner relies upon *Caron* for disclosure of "dependencies of procedural code objects stored in a database" and on *Colby* for disclosure of querying a database. The Examiner acknowledges, however, that neither *Caron* nor *Colby* disclose "recursively querying a database" and instead relies upon *Shan* for disclosure of the recited operation. (*Office Action*, page 5). Thus, to reject Applicants' step of "recursively querying a database for one or more dependencies of procedural objects stored in the database," the Examiner must piece together the disclosures of three separate references. Even if the cited references disclose the elements alleged by the Examiner (which Applicants do not admit), such a piecemeal rejection of Applicants' claim language fails to give credence to the particular combination of features recited in Applicants' claim. Applicants respectfully submit that a rejection of Claim 72 under the *Caron-Colby-Shan* combination, in the manner provided by the Examiner, can only result from piecing together disjointed portions of unrelated references to reconstruct Applicants' claims with the benefit of hindsight.

The Federal Circuit has made clear that is improper for an Examiner to use hindsight having read the Applicant's disclosure to arrive at an obviousness rejection. *In re* Fine, 837 F.2d 1071, 1075, 5 U.S.P.Q.2d 1596, 1600 (Fed. Cir. 1988). It is improper to use the claimed invention as an instruction manual or template to piece together the teachings of the prior art so that the claimed invention is rendered obvious. *In re* Fritch, 972 F.2d 1260, 23 U.S.P.Q.2d 1780 (Fed. Cir. 1992). In this case, Applicants respectfully submit that the Examiner has used Applicants' claimed invention as an instruction manual to combining the teachings of *Caron*, *Colby*, *Shan*, and *Laursen* to conclude that the references disclose "recursively querying a database for one or more dependencies of procedural code objects stored in the database." Because the hindsight reconstruction of Applicants' claim is improper, Applicants submit that Claim 72 is allowable over the proposed *Caron-Colby-Shan-Laursen* combination.

Finally, Applicants respectfully submit that one of ordinary skill in the art would not have been motivated to combine the disclosure of *Laursen* with that of *Caron*, *Colby*, and *Shan*. The question raised under 35 U.S.C. § 103 is whether the prior art taken as a whole would suggest the claimed invention taken as a whole to one of ordinary skill in the art at the time of the invention. "The mere fact that the references can be combined or modified does not render the resultant combination obvious unless the prior art also suggests the **desirability** of the combination." See M.P.E.P. §2143.01 (III) (emphasis original and added). Most recently, this requirement has been reaffirmed in an official USPTO memorandum dated May 3, 2007 wherein the Deputy Commissioner for Patent Operations pointed to sections of *KSR v. Teleflex*, which recite, "it will be necessary . . . to determine whether there was an **apparent reason** to combine the known elements in the fashion claimed by the patent at issue."[1]

In the instant case, the Examiner speculates "it would have been obvious" to make the proposed combination "for selecting, collecting, retrieving, and delivering information over the network in real-time or non-real-time (Laursen's col. 1, lines 10-10 and col. 2, lines 1-12." (*Office Action*, page 14). However, the Examiner's assertion ignores the purposes and objectives of *Laursen.* The Federal Circuit has stated that it is essential to view the invention as a whole, taking each element into account as well as the advantages, properties, utilities, and results of the invention. *In re Chupp,* 816 F.2d 643, 2 U.S.P.Q.2d 1437 (Fed. Cir. 1987).

[1] *KSR Int'l. Co v. Teleflex Inc.*, No. 04-1350 (April 30, 2007) (emphasis added).

As discussed above, *Caron* provides "a compiler and compiling method [that] minimizes recompilation of a computer program after an edit." (*Caron*, Abstract). Thus, the principle and purpose of the system disclosed in *Caron* is to decompile units "to one or more various intermediate compilation states according to the dependency of the dependent unit on the edited unit" such that "[w]hen next compiling the program, the units are compiled from their respective intermediate compilation state." (*Caron*, Abstract). In non-analogous art, *Laursen* discloses a "service mechanism [that] allows applications to be split such that client devices (set-top boxes, personal digital assistants, etc.) can focus on presentation, while backend services running in a distributed server complex, provide access to data via messaging across an abstracted interface." (*Laursen*, Abstract). Thus, the principles and objectives of the service used by set-top boxes and personal digital assistants disclosed in *Laursen* are vastly different than those of the computer program decompiler disclosed in *Caron*. Because these are non-analogous references with no relation to one another, Applicants respectfully submit that one of ordinary skill in the art at the time of invention would not have been motivated to combine the disclosure of *Caron* with the disclosure of *Laursen*.

The Federal Circuit has made clear that is improper for an Examiner to use hindsight having read the Applicant's disclosure to arrive at an obviousness rejection. *In re* Fine, 837 F.2d 1071, 1075, 5 U.S.P.Q.2d 1596, 1600 (Fed. Cir. 1988). It is improper to use the claimed invention as an instruction manual or template to piece together the teachings of the prior art so that the claimed invention is rendered obvious. *In re* Fritch, 972 F.2d 1260, 23 U.S.P.Q.2d 1780 (Fed. Cir. 1992). In this case, Applicants respectfully submit that the Examiner has used Applicants' claimed invention as an instruction manual to combining the teachings of *Caron*, *Colby*, *Shan*, and *Laursen* to conclude that the references disclose "for each of the one or more dependencies identified, determining whether the dependency already occurs in the graph" and "terminating the recursive query of the database upon determining that one of the one or more dependencies already occurs in the graph," as recited in Claim 72. Because the hindsight reconstruction of Applicants' claim is improper, Applicants submit that Claim 72 is allowable over the proposed *Caron-Colby-Shan-Laursen* combination.

For at least these reasons, Applicants respectfully request reconsideration and allowance of Claim 72.

C. **Claims 41, 61, and 66**

Dependent Claims 41, 61, and 66 depend on Claims 40, 60, and 65, respectively, which Applicants have shown above to be allowable. Accordingly, dependent Claims 41, 61, and 66 are not obvious over the proposed *Caron-Colby-Shan* combination at least because Claims 41, 61, and 66 include the limitations of their respective independent claims.

Additionally, Claims 41, 61, and 66 are patentable because they recite additional features and operation not disclosed, taught, or suggested in the proposed *Caron-Colby-Shan* combination. For example, the proposed *Caron-Colby-Shan* combination does not disclose, teach, or suggest steps analogous to both "recursively querying a database for one or more dependencies of **procedural code objects** stored in the database" and "recursively querying the database for one or more dependencies of **specifications of object-oriented code objects** stored in the database," as recited in Claim 41. In the *Office Action*, the Examiner relies on *Caron*, specifically, for disclosure of both "dependencies of procedural code objects" and "dependencies of specifications of object-oriented code objects" as recited in Applicants' claim 41. (*Office Action*, page 6). Applicants respectfully disagree, however, with this characterization of the disclosure of *Caron*.

Although *Caron* discloses that "three different types of intermodule dependencies are considered: layout, frame, and code" (Column 6, lines 28-30), none of these are analogous to "**one or more dependencies of specifications of object-oriented code objects**," as recited in Applicants' Claim 41. According to *Caron*, "[a] module has a layout dependency if its layout in object code (i.e., the size of storage allocated for each instance of the module at compile time and, in addition, the offset of members within the distance) depends on one or more statements in another module." (Column 6, lines 30-34). "Frame dependencies are commonly created by statements declaring local variables of a procedure" (Column 6, lines 51-52), and "[c]ode dependencies are commonly created by procedure calls, and other statements for which the amount of static or dynamic storage of the module is not altered by a change to another module, but for which a static address may be altered." (Column 7, lines 4-8). Thus, *Caron* merely discloses dependencies of statements and procedure calls. None of these are disclosed to be "dependencies of specifications of object-oriented code objects." Accordingly, Applicants submit that the proposed *Caron-Colby-Shan* combination, as relied upon by the Examiner, does not disclose, teach, or suggest the combination of elements recited in Applicants' Claim 41.

Additionally, Applicants note that in the *Office Action* the Examiner relies upon *Caron* for disclosure of "dependencies of procedural code objects stored in a database" and "dependencies of specifications of object-oriented code objects" but on *Colby* for disclosure of querying a database. The Examiner further acknowledges that neither *Caron* nor *Colby* disclose "recursively querying a database" and instead relies upon *Shan* for disclosure of the recited operation. (*Office Action*, page 5). Thus, to reject a single one of Applicants' steps, the Examiner must piece together the disclosures of three separate references. Even if the cited references disclose the elements alleged by the Examiner (which Applicants do not admit), such a piecemeal rejection of Applicants' claim language fails to give credence to the particular combination of features recited in Applicants' claim. Applicants respectfully submit that a rejection of Claim 40 under the *Caron-Colby-Shan* combination, in the manner provided by the Examiner, can only result from piecing together disjointed portions of unrelated references to reconstruct Applicants' claims with the benefit of hindsight.

The Federal Circuit has made clear that is improper for an Examiner to use hindsight having read the Applicant's disclosure to arrive at an obviousness rejection. *In re* Fine, 837 F.2d 1071, 1075, 5 U.S.P.Q.2d 1596, 1600 (Fed. Cir. 1988). It is improper to use the claimed invention as an instruction manual or template to piece together the teachings of the prior art so that the claimed invention is rendered obvious. *In re* Fritch, 972 F.2d 1260, 23 U.S.P.Q.2d 1780 (Fed. Cir. 1992). In this case, Applicants respectfully submit that the Examiner has used Applicants' claimed invention as an instruction manual to combining the teachings of *Caron*, *Colby*, and *Shan* to conclude that the references disclose "recursively querying a database for one or more dependencies of procedural code objects stored in the database" and "recursively querying a database for one or more specifications of object-oriented code objects stored in the database." Because the hindsight reconstruction of Applicants' claim is improper, Applicants submit that Claim 40 is allowable over the proposed *Caron-Colby-Shan* combination.

For at least these reasons, Applicants respectfully request consideration and allowance of Claim 41. Dependent Claims 61 and 66, which contain certain claim elements that are analogous to those discussed above in Claim 41, are allowable for similar reasons. New dependent Claim 74 is also allowable for analogous reasons.

D. Claims 42, 62, and 67

Dependent Claims 42, 62, and 67 depend on Claims 40, 60, and 65, respectively, which Applicants have shown above to be allowable. Accordingly, dependent Claims 42, 62, and 67 are not obvious over the proposed *Caron-Colby-Shan* combination at least because Claims 42, 62, and 67 include the limitations of their respective independent claims.

Additionally, Claims 42, 62, and 67 are patentable because they recite additional features and operation not disclosed, taught, or suggested in the proposed *Caron-Colby-Shan* combination. For example, Claim 42 recites "recursively querying the database for one or more dependencies of **implementations of object-oriented code objects** stored in the database." In the *Office Action*, the Examiner relies on *Caron*, specifically, for disclosure of "dependencies of specifications of object-oriented code objects," as recited in Applicants' Claim 42, but does not identify any specific reference as disclosing "dependencies of implementations of object-oriented code objects." (*Office Action*, page 6). Applicants respectfully submit that the cited references do not disclose "implementations of object-oriented code objects," as recited in Applicants' Claim 42.

Although *Caron* discloses that "three different types of intermodule dependencies are considered: layout, frame, and code" (Column 6, lines 28-30), none of these are analogous to "**one or more dependencies of implementations of object-oriented code objects**," as recited in Applicants' Claim 42. According to *Caron*, "[a] module has a layout dependency if its layout in object code (i.e., the size of storage allocated for each instance of the module at compile time and, in addition, the offset of members within the distance) depends on one or more statements in another module." (Column 6, lines 30-34). "Frame dependencies are commonly created by statements declaring local variables of a procedure" (Column 6, lines 51-52), and "[c]ode dependencies are commonly created by procedure calls, and other statements for which the amount of static or dynamic storage of the module is not altered by a change to another module, but for which a static address may be altered." (Column 7, lines 4-8). Thus, *Caron* merely discloses dependencies of statements and procedure calls. None of the dependencies of *Caron* are disclosed to be analogous to "dependencies of implementations of object-oriented code objects." Additionally, because *Colby* merely relates to rewriting queries and *Shan* merely relates generally to translating recursive queries into algebraic statements, the additional disclosures of *Colby* and *Shan* do not cure the deficiencies of *Colby*. Accordingly, Applicants submit that the proposed *Caron-Colby-Shan*

combination does not disclose, teach, or suggest "one or more dependencies of implementations of object-oriented code objects," as recited in Applicants' Claim 42.

Additionally, Applicants note that in the *Office Action* the Examiner relies upon *Caron* for disclosure of "dependencies of procedural code objects stored in a database" and "dependencies of implementations of object-oriented code objects" but on *Colby* for disclosure of querying a database. The Examiner further acknowledges that neither *Caron* nor *Colby* disclose "recursively querying a database" and instead relies upon *Shan* for disclosure of the recited operation. (*Office Action*, page 5). Thus, to reject a single one of Applicants' steps, the Examiner must piece together the disclosures of three separate references. Even if the cited references disclose the elements alleged by the Examiner (which Applicants do not admit), such a piecemeal rejection of Applicants' claim language fails to give credence to the particular combination of features recited in Applicants' claim. Applicants respectfully submit that a rejection of Claim 40 under the *Caron-Colby-Shan* combination, in the manner provided by the Examiner, can only result from piecing together disjointed portions of unrelated references to reconstruct Applicants' claims with the benefit of hindsight.

The Federal Circuit has made clear that is improper for an Examiner to use hindsight having read the Applicant's disclosure to arrive at an obviousness rejection. *In re* Fine, 837 F.2d 1071, 1075, 5 U.S.P.Q.2d 1596, 1600 (Fed. Cir. 1988). It is improper to use the claimed invention as an instruction manual or template to piece together the teachings of the prior art so that the claimed invention is rendered obvious. *In re* Fritch, 972 F.2d 1260, 23 U.S.P.Q.2d 1780 (Fed. Cir. 1992). In this case, Applicants respectfully submit that the Examiner has used Applicants' claimed invention as an instruction manual to combining the teachings of *Caron*, *Colby*, and *Shan* to conclude that the references disclose "recursively querying a database for one or more dependencies of procedural code objects stored in the database" and "recursively querying a database for one or more dependencies of implementations of object-oriented code objects." Because the hindsight reconstruction of Applicants' claim is improper, Applicants submit that Claim 40 is allowable over the proposed *Caron-Colby-Shan* combination.

For at least these reasons, Applicants respectfully request consideration and allowance of Claim 42. Claims 62 and 67, which contain certain claim elements that are analogous to those discussed above in Claim 42, are allowable for similar reasons.

E. Claims 48, 49, and 74

Claim 48 has been rewritten in independent form to include the limitations recited in Claim 40 prior to any amendment in this Response to Office Action. Applicants respectfully submits that the proposed *Caron-Colby-Shan* combination does not disclose, teach, or suggest the each and every element recited in Applicant's Claim 48, as previously presented.

For example, the proposed *Caron-Colby-Shan* combination does not disclose, teach, or suggest "identifying one or more cyclic dependencies among code objects stored in the database," as recited in Applicants' Claim 48. In the *Office Action*, the Examiner continues to rely specifically on *Colby* for disclosure of the claim elements of Claim 48. Specifically, the Examiner cites column 7, lines 4-15 and column 10, lines 18-32 of *Colby*. Applicants contend, however, that neither cited section discloses the elements recited in Applicants' Claims 48 and 49.

The first cited portion of *Colby* discloses a system for answering a database query where the tables "include foreign key-primary key relationships." (Column 7, lines 4-8). According to *Colby*, the "primary key uniquely identifies each row in a database table" and "can be one value from a single column or a combination of values from multiple columns." (Column 7, lines 8-11). By contrast, "[a] foreign key column contains only the values of a primary key column of another table and establishes a many-to-one relationship between the two tables." (Column 7, lines 11-13). "Unlike a primary key column a foreign key column can contain duplicate values." Thus, the cited portion of *Colby* merely discloses that a primary key uniquely identifies each row in a table but that a foreign key establishes a many-to-one relationship between two tables. There is no disclosure in the cited portion that either of the foreign key or primary key are analogous to "one or more cyclic dependencies among code objects stored in the database," as recited in Applicants' Claim 48. Certainly, their is no disclosure in the cited portion of *Colby* of "utilizing a graph traversal algorithm to identify one or more cyclic dependencies," as recited in Claim 49.

The second cited portion of *Colby* discloses a derived table that is created in an example rewritten query. Specifically, *Colby* discloses:

> Note that if day was a primary key of the period table, then no derived table would be needed. A reason for creating the derived table of FIG. 5B is to provide an accurate result. If the sum_dollars_by_day table of FIG. 5A was simply joined with the period table of FIG. 4 under the column day, then the

> total sum of dollars for each day would be multiplied by the number of times that day appeared in the period of FIG. 4, rather than having the desired effect of having a single sum of the dollar transactions for a given day.

(Column 10, lines 10-26). Thus, *Colby* merely discloses that the primary key results in an accurate summation. It does not at all relate to "cyclic dependencies," as recited in Claims 48 and 49. Thus, this portion of *Colby* also does not disclose, teach, or suggest "identifying one or more cyclic dependencies among code objects stored in the database" or "utilizing a graph traversal algorithm to identify one or more cyclic dependencies," as recited in Claims 48 and 49, respectively.

For at least these reasons, Applicants respectfully request consideration and allowance of Claims 48. For analogous reasons, Applicants respectfully request reconsideration and allowance of new dependent Claim 74, which recites features analogous to Claim 48, and Claim 49, which recites "utilizing a graph traversal algorithm to identify one or more cyclic dependencies," as recited in Claim 49.

F. Claim 50

Dependent Claim 50 depends on Claims 40, which Applicants have shown above to be allowable. Accordingly, dependent Claim 50 is not obvious over the proposed *Caron-Colby* combination at least because Claim 50 includes the limitations of independent Claim 40.

Additionally, Claim 50 is patentable because they recite additional features and operation not disclosed, taught, or suggested in the proposed *Caron-Colby-Shan* combination. For example, the proposed *Caron-Colby-Shan* combination does not disclose, teach, or suggest "generating a dependency graph for code objects stored in the database based at least in part on the dependency information tracking array," as recited in Claim 50. In the *Office Action*, the Examiner states that *Colby* "teaches . . . a dependency graph . . . based at least in part on the dependency information tracking array." (*Office Action*, page 8). Applicants respectfully disagree. The cited portion of *Colby* merely discloses tables that "include foreign key-primary key relationships." (Column 7, lines 4-8). According to *Colby*, the "primary key uniquely identifies each row in a database table" and "can be one value from a single column or a combination of values from multiple columns." (Column 7, lines 8-11). By contrast, "[a] foreign key column contains only the values of a primary key column of

another table and establishes a many-to-one relationship between the two tables." (Column 7, lines 11-13). "Unlike a primary key column a foreign key column can contain duplicate values." (Column 7, lines 13-14). According to *Colby*, the primary key results in an accurate summation." (Column 10, lines 10-26). However, the use of primary key and foreign key relationships is not analogous to "generating a dependency graph for code objects stored in the database based at least in part on the dependency information tracking array," as recited in Applicants' Claim 50.

For at least these reasons, Applicants respectfully request consideration and allowance of Claim 50.

G. Claims 70 and 71

Dependent Claims 70 and 71 depend, either directly or indirectly, on Claim 40, which Applicants have shown above to be allowable. Accordingly, dependent Claims 70 and 71 are not obvious over the proposed *Caron-Colby-Shan-Laursen* combination at least because Claims 70 and 71 include the limitations of independent Claim 40 and any intervening claims.

Additionally, Claims 70 and 71 are patentable because they recite additional features and operation not disclosed, taught, or suggested in the proposed *Caron-Colby-Shan-Laursen* combination. For example, the proposed *Caron-Colby-Shan-Laursen* combination does not disclose, teach, or suggest that "the specifications of object-oriented code objects comprise PL/SQL specifications for a collection of stored functions and procedures identified as a single entity," as recited in Claim 70. In the *Office Action*, the Examiner acknowledges that *Caron*, *Colby*, and *Shan* do not disclose the recited claim elements. (*Office Action*, page 13). Instead, the Examiner relies on *Laursen* for disclosure of the recited claim elements. Specifically, the Examiner states that "*Laursen* teaches PL/SQL statements (dext 59)." (*Office Action*, page 13). Again, Applicants are not sure what "dext" refers to and have determined that there is no reference in *Laursen* to the reference numeral 59. It does not appear that 59 refers to a page number, a line number, or a paragraph number. In short, the Examiner's rejection of the claim language leaves Applicants' guessing as to the basis for the Examiner's rejection and provides the Applicants with little guidance in how to respond.

Furthermore, Applicants have performed a review of *Laursen* and have concluded that *Laursen* does not disclose, teach, or suggest that "the specifications of object-oriented code

objects comprise PL/SQL specifications for a collection of stored functions and procedures identified as a single entity," as recited in Claim 70. In fact, the only reference to PL/SQL in *Laursen* states that "remote procedure calls (RPC) to access services and data through the media server 100 is preferred over the use of a traditional Structured Query Language (SQL) for data access." (*Laursen*, Column 7, lines 45-49). According to *Laursen*, "[t]his is because it reduces the number of round-trip messages and provides easy to use interfaces to application services." (*Laursen*, Column 7, lines 49-51). Thus, *Laursen* actually teaches away from the use of SQL statements. Certainly there is no disclosure in *Laursen* that "the specifications of object-oriented code objects comprise PL/SQL specifications for a collection of stored functions and procedures identified as a single entity," as recited in Claim 70.

Finally, Applicants respectfully submit that one of ordinary skill in the art would not have been motivated to combine the disclosure of *Laursen* with that of *Caron*, *Colby*, and *Shan*. The question raised under 35 U.S.C. § 103 is whether the prior art taken as a whole would suggest the claimed invention taken as a whole to one of ordinary skill in the art at the time of the invention. "The mere fact that the references can be combined or modified does not render the resultant combination obvious unless the prior art also suggests the **desirability** of the combination." See M.P.E.P. §2143.01 (III) (emphasis original and added). Most recently, this requirement has been reaffirmed in an official USPTO memorandum dated May 3, 2007 wherein the Deputy Commissioner for Patent Operations pointed to sections of *KSR v. Teleflex*, which recite, "it will be necessary . . . to determine whether there was an **apparent reason** to combine the known elements in the fashion claimed by the patent at issue."[2]

In the instant case, the Examiner speculates "it would have been obvious" to make the proposed combination "for selecting, collecting, retrieving, and delivering information over the network in real-time or non-real-time (Laursen's col. 1, lines 10-10 and col. 2, lines 1-12." (*Office Action*, pages 13-14). However, the Examiner's assertion ignores the purposes and objectives of *Laursen*. The Federal Circuit has stated that it is essential to view the invention as a whole, taking each element into account as well as the advantages, properties, utilities, and results of the invention. *In re Chupp*, 816 F.2d 643, 2 U.S.P.Q.2d 1437 (Fed. Cir. 1987). As discussed above, *Caron* provides "a compiler and compiling method [that]

[2] *KSR Int'l. Co v. Teleflex Inc.*, No. 04-1350 (April 30, 2007) (emphasis added).

minimizes recompilation of a computer program after an edit." (*Caron*, Abstract). Thus, the principle and purpose of the system disclosed in *Caron* is to decompile units "to one or more various intermediate compilation states according to the dependency of the dependent unit on the edited unit" such that "[w]hen next compiling the program, the units are compiled from their respective intermediate compilation state." (*Caron*, Abstract). In non-analogous art, *Laursen* discloses a "service mechanism [that] allows applications to be split such that client devices (set-top boxes, personal digital assistants, etc.) can focus on presentation, while backend services running in a distributed server complex, provide access to data via messaging across an abstracted interface." (*Laursen*, Abstract). Thus, the principles and objectives of the service used by set-top boxes and personal digital assistants disclosed in *Laursen* are vastly different than those of the computer program decompiler disclosed in *Caron*. Because these are non-analogous references with no relation to one another, Applicants respectfully submit that one of ordinary skill in the art at the time of invention would not have been motivated to combine the disclosure of *Caron* with the disclosure of *Laursen*.

The Federal Circuit has made clear that is improper for an Examiner to use hindsight having read the Applicant's disclosure to arrive at an obviousness rejection. *In re* Fine, 837 F.2d 1071, 1075, 5 U.S.P.Q.2d 1596, 1600 (Fed. Cir. 1988). It is improper to use the claimed invention as an instruction manual or template to piece together the teachings of the prior art so that the claimed invention is rendered obvious. *In re* Fritch, 972 F.2d 1260, 23 U.S.P.Q.2d 1780 (Fed. Cir. 1992). In this case, Applicants respectfully submit that the Examiner has used Applicants' claimed invention as an instruction manual to combining the teachings of *Caron*, *Colby*, *Shan*, and *Laursen* to conclude that the references disclose that "the specifications of object-oriented code objects comprise PL/SQL specifications for a collection of stored functions and procedures identified as a single entity," as recited in Claim 70. Because the hindsight reconstruction of Applicants' claim is improper, Applicants submit that Claim 70 is allowable over the proposed *Caron-Colby-Shan-Laursen* combination.

For at least these reasons, Applicants respectfully request reconsideration and allowance of Claim 70. For analogous reasons, Applicants respectfully request reconsideration and allowance of Claim 71.

H. Claim 73

Dependent Claim 73 depends on Claim 40, which Applicants have shown above to be allowable. Accordingly, dependent Claim 73 is not obvious over the proposed *Caron-Colby-Shan-Laursen* combination at least because Claim 73 includes the limitations of independent Claim 40.

Additionally, Claim 73 are patentable because they recite additional features and operation not disclosed, taught, or suggested in the proposed *Caron-Colby-Shan-Laursen* combination. For example, the proposed *Caron-Colby-Shan-Laursen* combination does not disclose, teach, or suggest that "displaying a dependency graph to a user, the dependency graph generated based at least in part on the dependency information tracking array," as recited in Claim 73. In the *Office Action*, the Examiner acknowledges that *Caron*, *Colby*, and *Shan* do not disclose the recited claim elements. (*Office Action*, page 14). Instead, the Examiner relies on *Laursen* for disclosure of the recited claim elements. Specifically, the Examiner states that "*Laursen* teaches directed graph and tracking arrays (dext 99, 101, 121, and 125)." (*Office Action*, page 14). Thus, the Examiner identifies only that *Laursen* relates to directed graph and tracking arrays but does not identify that *Laursen* actually teaches "displaying a dependency graph to a user, the dependency graph generated based at least in part on the dependency information tracking array," as recited in Claim 73. Again, Applicants are not sure what "dext" refers to and have determined that there is no use in *Laursen* of the reference numerals 99, 101, 121, and 125. It does not appear that the numerals provided by the Examiner refer to page numbers, line numbers, or paragraph numbers. In short, the Examiner's rejection of the claim language leaves Applicants' guessing as to the basis for the Examiner's rejection and provides Applicants with little guidance in how to respond.

Furthermore, Applicants have performed a review of *Laursen* and have concluded that *Laursen* does not at all relate to dependency graphs or to dependency information tracking arrays. *Laursen* merely discloses:

> The network protocol of the present invention is composed on one or more underlying networks, each with their own well known characteristics. In the description that follows, the topology of the network is described as a directed graph whose vertices are nodes (either intermediate nodes or endpoints) and whose edges are data links between nodes.

(*Laursen*, Column 12, lines 40-46). Thus, while *Laursen* discloses a "directed graph", the graph is a topology of a network with intersections of the graph including nodes of the network. There is no indication in *Laursen* that the directed graph is analogous to a dependency graph or a dependency information tracking array. Certainly, there is no disclosure in *Laursen* of "displaying a dependency graph to a user, **the dependency graph generated based at least in part on the dependency information tracking array**," as recited in Claim 73. Accordingly, Claim 73 is allowable over the proposed *Caron-Colby-Shan-Laursen* combination.

Finally, Applicants respectfully submit that one of ordinary skill in the art would not have been motivated to combine the disclosure of *Laursen* with that of *Caron*, *Colby*, and *Shan*. The question raised under 35 U.S.C. § 103 is whether the prior art taken as a whole would suggest the claimed invention taken as a whole to one of ordinary skill in the art at the time of the invention. "The mere fact that the references can be combined or modified does not render the resultant combination obvious unless the prior art also suggests the **desirability** of the combination." See M.P.E.P. §2143.01 (III) (emphasis original and added). Most recently, this requirement has been reaffirmed in an official USPTO memorandum dated May 3, 2007 wherein the Deputy Commissioner for Patent Operations pointed to sections of *KSR v. Teleflex*, which recite, "it will be necessary . . . to determine whether there was an **apparent reason** to combine the known elements in the fashion claimed by the patent at issue."[3]

In the instant case, the Examiner speculates "it would have been obvious" to make the proposed combination "for selecting, collecting, retrieving, and delivering information over the network in real-time or non-real-time (Laursen's col. 1, lines 10-10 and col. 2, lines 1-12." (*Office Action*, page 14). However, the Examiner's assertion ignores the purposes and objectives of *Laursen*. The Federal Circuit has stated that it is essential to view the invention as a whole, taking each element into account as well as the advantages, properties, utilities, and results of the invention. *In re Chupp,* 816 F.2d 643, 2 U.S.P.Q.2d 1437 (Fed. Cir. 1987). As discussed above, *Caron* provides "a compiler and compiling method [that] minimizes recompilation of a computer program after an edit." (*Caron*, Abstract). Thus, the principle and purpose of the system disclosed in *Caron* is to decompile units "to one or more various intermediate compilation states according to the dependency of the dependent unit on the

[3] *KSR Int'l. Co v. Teleflex Inc.*, No. 04-1350 (April 30, 2007) (emphasis added).

edited unit" such that "[w]hen next compiling the program, the units are compiled from their respective intermediate compilation state." (*Caron*, Abstract). In non-analogous art, *Laursen* discloses a "service mechanism [that] allows applications to be split such that client devices (set-top boxes, personal digital assistants, etc.) can focus on presentation, while backend services running in a distributed server complex, provide access to data via messaging across an abstracted interface." (*Laursen*, Abstract). Thus, the principles and objectives of the service used by set-top boxes and personal digital assistants disclosed in *Laursen* are vastly different than those of the computer program decompiler disclosed in *Caron*. Because these are non-analogous references with no relation to one another, Applicants respectfully submit that one of ordinary skill in the art at the time of invention would not have been motivated to combine the disclosure of *Caron* with the disclosure of *Laursen*.

The Federal Circuit has made clear that is improper for an Examiner to use hindsight having read the Applicant's disclosure to arrive at an obviousness rejection. *In re* Fine, 837 F.2d 1071, 1075, 5 U.S.P.Q.2d 1596, 1600 (Fed. Cir. 1988). It is improper to use the claimed invention as an instruction manual or template to piece together the teachings of the prior art so that the claimed invention is rendered obvious. *In re* Fritch, 972 F.2d 1260, 23 U.S.P.Q.2d 1780 (Fed. Cir. 1992). In this case, Applicants respectfully submit that the Examiner has used Applicants' claimed invention as an instruction manual to combining the teachings of *Caron*, *Colby*, *Shan*, and *Laursen* to conclude that the references disclose "displaying a dependency graph to a user, the dependency graph generated based at least in part on the dependency information tracking array," as recited in Claim 73. Because the hindsight reconstruction of Applicants' claim is improper, Applicants submit that Claim 73 is allowable over the proposed *Caron-Colby-Shan-Laursen* combination.

For at least these reasons, Applicants respectfully request reconsideration and allowance of Claim 73.

No Waiver

Additionally, Applicants have merely discussed example distinctions from the references cited by the Examiner. Other distinctions may exist, and Applicants reserve the right to discuss these additional distinctions in a later Response or on Appeal, if appropriate. By not responding to additional statements made by the Examiner, Applicants do not acquiesce to the Examiner's additional statements. The example distinctions discussed by Applicants are sufficient to overcome the Examiner's rejections.

CONCLUSION

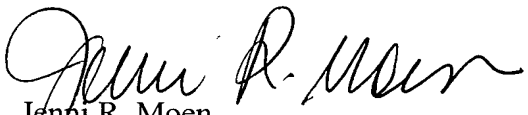
Applicants have made an earnest attempt to place this case in condition for allowance. For the foregoing reasons, and for other apparent reasons, Applicants respectfully request full allowance of all pending Claims. If the Examiner feels that a telephone conference or an interview would advance prosecution of this Application in any manner, the undersigned attorney for Applicants stands ready to conduct such a conference at the convenience of the Examiner.

The Commissioner is authorized to charge the amount of $470.00 for additional claims to Deposit Account No. 02-0384 of Baker Botts L.L.P. Although no other fees are believed due, please credit any overpayment or charge any additional fee required by this paper to Deposit Account No. 02-0384 of Baker Botts L.L.P.

Respectfully submitted,

BAKER BOTTS L.L.P.

Attorneys for Applicants



Jenni R. Moen
Reg. No. 52,038
(214) 953-6809

Date: March 27, 2008

CORRESPONDENCE ADDRESS:

Customer Number: **05073**